Inteligência Artificial - Grupo: 02
André Fonseca - 84698
Leonor Loureiro - 84736

TÉCNICO LISBOA

Aprendizagem Automática
Inteligência Artificial 2017/2018

P1 - Métodos de Classificação

Escolha de Features

Para nos auxiliar no processo de escolha das features utilizámos o modelo *ExtraTreesClassifier*. O método *feature_importances_* deste modelo retorna a importância relativa de cada atributo. Um valor mais elevado para um determinado atributo indica que este é mais relevante para o processo de classificação.

Feature	Importância Relativa
len(X[x])	0,33241987
ord(X[x][0])	0,251637885
nr_vogais(X[x])	0,064413275
ord(X[x][-1])	0,18797154
hash(X[x])	0,16355743

Método Aprendizagem

Escolhemos o modelo *KNeighborsClassifier* para classificar um conjunto de palavras. O princípio base deste método passa por encontrar um número predefinido de amostras do caso de treino mais próximas do novo ponto, e predizer a classificação do novo ponto a partir da classificação destas amostras.

Escolha e Análise de Parâmetros

Para avaliarmos a qualidade da predição relativamente a diferentes valores para os parâmetros, executámos uma *GridSearchCV*, um método de validação cruzada, que avalia todas as combinações possíveis de uma grelha de valores, para alguns parâmetros. Como métrica, utilizamos *f1*.

Precision é o rácio entre o número amostras corretamente classificadas positivas e a soma do número total de amostras classificadas como positivas.

$$precicion = \frac{tp}{tp + fp}$$

Recall avalia a capacidade de um classificador identificar corretamente todos as amostras positivas. É o rácio entre o número de amostras corretamente classificadas como positiva e o número total de amostras realmente positivas.

$$recall = \frac{tp}{tp + fp}$$

F1 (F-measure) é a média pesada das métricas recall e precision. O seu melhor valor é 1 e o pior é 0.

$$f1 = \frac{2(precision * recall)}{precision + recall}$$

		wordclass	wordclass2
weights	n_neighbors	f1	f1
uniform	1	0,493	0,517
distance	1	0,493	0,517
uniform	2	0,448	0,492
distance	2	0,493	0,517
uniform	3	0,49	0,506
distance	3	0,512	0,51
uniform	4	0,456	0,498
distance	4	0,51	0,511
uniform	5	0,484	0,442
distance	5	0,478	0,494

Os parâmetros que apresentam os melhores resultados para os dois exemplos são `weights = 'distance'` e `n_neighbors = 3`.

P2 - Métodos de Regressão

Métodos de Aprendizagem

Escolhemos os modelos Kernel Ridge e SVR. O Kernel Ridge combina a ridge regression com o kernel trick, que calcula os produtos internos de vetores num espaço dimensional maior $\mathbb{R}^m$ sem sair do espaço $\mathbb{R}^n$. O modelo SVR cria um hiperplano de dimensão muito elevada, que utiliza para calcular a regressão.

Análise e Escolha de Parâmetros

Para avaliarmos a qualidade da predição relativamente a diversos valores dos parâmetros, recorremos novamente ao método *GridSearchCV*, desta vez usando a métrica *neg_mean_squared_error*, que corresponde ao simétrico do valor esperado do erro quadrático médio. O melhor valor possível desta métrica é 0.

SVR			
C	gamma	regress.npy	regress2.npy
1000	0,0001	-1,107	-1585,388
1000	0,001	-2,495	-1316,274
1000	0,01	-5,118	-2959,286
1000	0,1	-0,161	-582,911
1000	1	-0,415	-2931,746
10000	0,0001	-1,38	-1228,673
10000	0,001	-2,906	-1689,798
10000	0,01	-3,151	-88,226
10000	0,1	-2,964	-257,903
10000	1	-0,415	-2931,746

KernelRidge				
kernel	**alpha**	**gamma**	**regress.npy**	**regress2.npy**
rbf	0.01	1.0	-1,672	-1952,743
polynomial	0.01	1.0	-8,885	-0,103
rbf	0.01	0.1	-0,1	-811,083
polynomial	0.01	0.1	-7,305	-0,172
rbf	0.01	0.01	-1,958	-2232,32
polynomial	0.01	0.01	-2,982	-2302,763
rbf	0.01	0.001	-1,033	-979,015
polynomial	0.01	0.001	-1,088	-990,19
rbf	0.001	1.0	-1,689	-1940,123
polynomial	0.001	1.0	-11,605	-0,106
rbf	0.001	0.1	-0,096	-547,943
polynomial	0.001	0.1	-11,206	-0,098
rbf	0.001	0.01	-3,946	-582,056
polynomial	0.001	0.01	-4,674	-545,993
rbf	0.001	0.001	-1,653	-1621,042
polynomial	0.001	0.001	-1,885	-1877,724

Os parâmetros que apresentam os melhores resultados são:

***KernelRidge*:**
`alpha=0.001,gamma=0.1,kernel='rbf'`

***SVR*:**
`kernel='rbf',C=1000,gamma=0.1`

O método *KernelRidge* apresenta um erro quadrático médio inferior, pelo que representa a melhor predição.

P3 - Aprendizagem por reforço

Traces2Q
Função que gera matriz Q de valores, que corresponde à implementação do QLearning.

Q2pol
Função de exploração/policy baseada no ε greedy que tem um fator aleatório associado à escolha da acção, dando prioridade à com maior Q value.

Trajetória gerada nos ambientes testados:
Ambiente 1 [state, action, next state, reward]:
[[5. 0. 6. 0.] [6. 0. 6. 1.] [6. 0. 6. 1.] [6. 0. 6. 1.]]
Ambiente 2 [state, action, next state, reward]:
[[5. 0. 5. 0.] [5. 0. 6. 0.] [6. 0. 1. 1.] [1. 1. 0. 0.]]

Função recompensa:
Função recompensa R, que dado um estado inicial **s** e uma acção **a** devolve a recompensa de executar **a** em **s:**

$$R(s, a) \begin{cases} \textbf{1}, \text{ se } s = 0 \text{ v } s = 6 \\ \textbf{0}, \text{ caso contrário} \end{cases}$$

Movimento do Agente:
O agente move-se pela trajetória gerada com a política e valores Q aprendidos relativos a cada ambiente, por exemplo nos ambientes testados andou pelos seguintes estados (que podem ser vistos na representação gráfica):
1º Ambiente: 5 -> 6 -> 6 -> 6
2º Ambiente: 5 -> 5 -> 6 -> 1

Representação gráfica dos ambientes: